Python

Strings and Formatting

@kabirbaidhya

Reflections

What we know now

After the previous sessions we know about the following:

1. Variables in Python
2. Data Types
3. Operators and Expressions
4. Git - a little bit more :)

Strings

String

A string is traditionally a sequence of characters.

Strings are one of the common data types in all programming languages and python is not an exception.

String in Python is handled with `str` object and strings are immutable sequences.

Strings in Python

Strings can be represented in various of ways:

```
# Using Single Quotes
my_string1 = 'This is a string'

# Using Double Quotes
my_string2 = "This is a string too".

# Using Triple Quotes (Multiline strings)
my_string3 = '''Lorem ipsum dolor sit amet, consectetur adi
sed do eiusmod tempor incididunt ut labore et dolore magna

# You could write it with double quotes as well
my_string3 = """Lorem ipsum dolor sit amet, consectetur adi
sed do eiusmod tempor incididunt ut labore et dolore magna
```

Common Operations

The following operations are supported by strings and most of the sequences.

Operation	Result
x in s	True if an item of s is equal to x, else False
x not in s	False if an item of s is equal to x, else True
s + t	the concatenation of s and t
s * n or n * s	equivalent to adding s to itself n times
s[i]	ith item of s, origin 0
s[i:j]	slice of s from i to j
s[i:j:k]	slice of s from i to j with step k

Common Operations

Operation	Result
len(s)	length of s
min(s)	smallest item of s
max(s)	largest item of s
s.index(x[, i[, j]])	index of the first occurrence of x in s (at or after index i and before index j)
s.count(x)	total number of occurrences of x in s

Example 1

```
s = input('Enter a string: ')

print("You have entered " + s)
print("No. of characters = %d" % len(s))
print("First Character = %s" % s[0])
print("Last Character = %s" % s[len(s) - 1])
```

Example 2

```
s = input('Enter a string: ')

# Count the number of vowels
print("No. of 'a' = %s" % s.count('a'))
print("No. of 'e' = %s" % s.count('e'))
print("No. of 'i' = %s" % s.count('i'))
print("No. of 'o' = %s" % s.count('o'))
print("No. of 'u' = %s" % s.count('u'))

# Calculate Percentage of vowels
total_vowels = s.count('a') + s.count('e') + s.count('i') +
percentage = (float(total_vowels) / len(s)) * 100
print("\n%.2f%% are vowels." % percentage)
```

String Methods

Method	Description
capitalize()	Return a new string with its first character capitalized and the rest lowercased.
endswith(suffix[, start[, end])	Check if the string ends with the given suffix. Return boolean result `True` or `False`
startswith(prefix[, start[, end]])	Check if the string starts with the given prefix and return boolean result `True` or `False`
find(sub[, start[, end]])	Return the first index in the string where substring `sub` is found within `start` to `end` of the string. Return -1 if not found.

String Methods

Method	Description
strip([chars])	Return a new string with the leading and trailing characters removed. The optional `chars` argument defaults to removing whitespace.
swapcase()	Return a new string with uppercase characters converted to lowercase and vice versa.
title()	Return a new titlecased version of the string where words start with an uppercase character and other letters are lowercased.
upper()	Return a new uppercased version of the string.

String Methods

Method	Description
lower()	Return a new lowercased version of the string.
split(sep=None, maxsplit=-1)	Splits the string into substring using the `sep` argument as the separator. Return the list of splitted substrings.
format(*args, **kwargs)	Perform a string formatting operation and return the formatted string.
replace(old, new[, count])	Return a new string by replacing all occurrences of substring `old` with `new`. If `count` argument is provided, only `count` number of replacements would be done.

Example 3

```
text = input('Enter a string: ')

print("capitalize()       = ", text.capitalize())
print("strip()            = ", text.strip())
print("swapcase()         = ", text.swapcase())
print("title()            = ", text.title())
print("upper()            = ", text.upper())
print("lower()            = ", text.lower())
print("replace('a', 'b') = ", text.replace('a', 'b'))
print("endswith('foo')    = ", text.endswith('foo'))
print("startswith('bar') = ", text.startswith('bar'))
print("find('foo')        = ", text.find('foo'))
print("split(' ')         = ", text.split(' '))
```

C-Style Formatting

C-Style formatting

You probably remember the `printf` function if you've programmed in C. You can do similar string formatting in Python as well.

You would do something like this.

```
print("Hello %s!" % name)
```

Example 4

```
# Ask the user to enter first and last name.
first_name = input('Your first name: ')
last_name = input('Your last name: ')

print("\nHi %s %s!" % (first_name, last_name))
print("It's nice to meet you.")
```

Example 5

```
# Ask the user to enter first and last name.
PI = 3.1415
radius = input('Enter radius of circle(meters): ')
area = PI * float(radius) ** 2

print("\nArea of circle = %.2f sq. metres" % area)
```

Format specifiers

Following are the supported conversion types.

Conversion	Meaning
'd'	Signed integer decimal.
'i'	Signed integer decimal.
'o'	Signed octal value.
'u'	Obsolete type – it is identical to 'd'.
'x'	Signed hexadecimal (lowercase).
'X'	Signed hexadecimal (uppercase).
'e'	Floating point exponential format (lowercase).
'E'	Floating point exponential format (uppercase).

Format specifiers

Conversion	Meaning
'f'	Floating point decimal format.
'F'	Floating point decimal format.
'g'	Floating point format. Uses lowercase exponential format if exponent is less than -4 or not less than precision, decimal format otherwise.
'G'	Floating point format. Uses uppercase exponential format if exponent is less than -4 or not less than precision, decimal format otherwise.
'c'	Single character (accepts integer or single character string).

Format specifiers

Conversion	Meaning
'r'	String (converts any Python object using repr()).
's'	String (converts any Python object using str()).
'a'	String (converts any Python object using ascii()).
'%'	No argument is converted, results in a '%' character in the result.

For in-depth information about the C-style formatting check the official docs.

New style formatting

New style formatting

Python provides another way for formatting as well.

That is using `str.format()` method.

Something like this:

```
print("Hello {}!".format(name))
```

Pretty much the same, right?

Example 6

Okay, check this example on what difference this new syntax makes.

```
first_name = input('Your first name: ')
last_name = input('Your last name: ')

# Old style formatting.
print('Hello %s %s!' % (first_name, last_name))

# New Style formatting
print('Hello {} {}!'.format(first_name, last_name))
print('Hello {0} {1}!'.format(first_name, last_name))

# This is where, you will feel the difference.
print('Hello {1} {0}!'.format(first_name, last_name))
print('Hello {0} {0} {1}!'.format(first_name, last_name))
```

Example 7

It supports all the format specifiifiers you've used in C-Style style formatting.

Check this.

```
amount = input('Enter amount in USD: ')
rate = 100.00

amount_npr = float(amount) * rate
print('Equivalent amount: NPR. {:.2f}'.format(amount_npr))
```

Exercises

Exercise 1

Write a program to ask for the marks of 5 different subjects and print the total marks obtained and the total percentage.

Exercise 2

Write a program to ask for the equation of a line in the form `y = mx + c`. And print the values of slope and y-intercept of the line. (Hint: Use `split()`.)

Exercise 3

Write a program to ask for the user's date of birth in `YYYY-MM-DD` format and calculate the user's age. (Hint: Use `split()` method.)

Read More?

Links

1. https://docs.python.org/3/library/stdtypes.html#old-string-formatting
2. https://pyformat.info/
3. https://www.tutorialspoint.com/python/python_strings.htm

Thank You

@kabirbaidhya

kabirbaidhya@gmail.com